Distributed Computing Grid Experiences in CMS Data Challenge

A. Fanfani

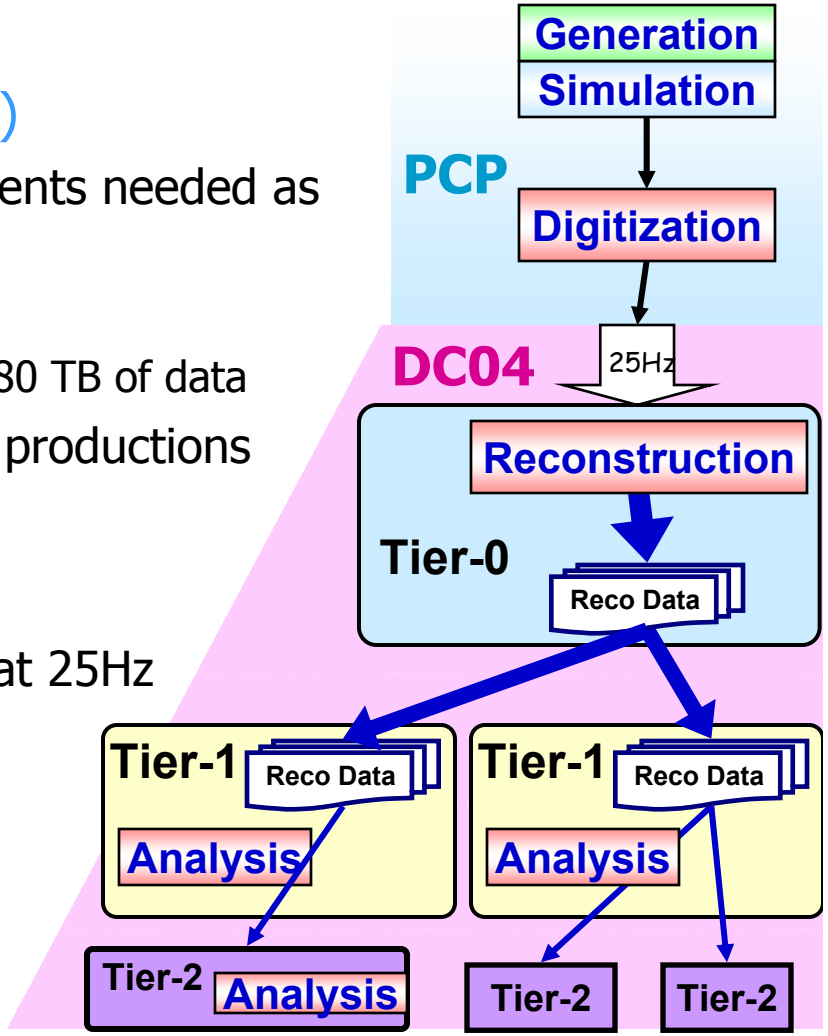
Dept. of Physics and INFN, Bologna

on behalf of the CMS Collaboration

CMS Data Challenge 2004

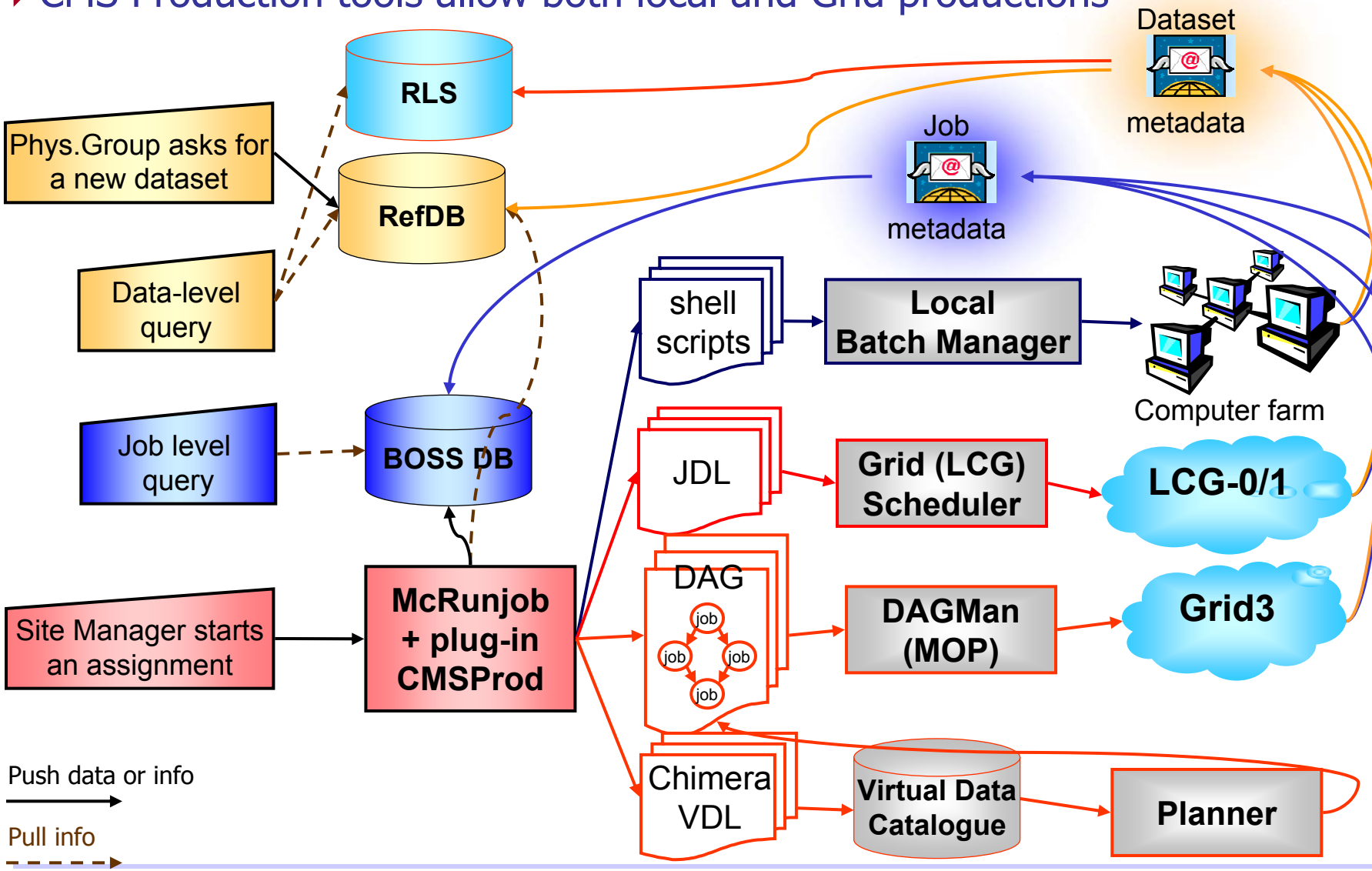
Planned to reach a complexity scale equal to about 25% of that foreseen for LHC initial running

- Pre-Challenge Production in 2003/04 (PCP)
  - Simulation and digitization of ~70 Million events needed as input for the Data Challenge
    - started in July 2003, Digitization still running
    - 750K jobs, 3500 KSI2000 months, 700 Kfiles,80 TB of data
  - Classic and Grid (CMS/LCG-0, LCG-1, Grid3) productions
- Data Challenge (DC04)
  - Reconstruction of data for sustained period at 25Hz
  - Data distribution to Tier-1,Tier-2 sites
  - Data analysis at remote sites
  - Demonstrate the feasibility of the full chain



Pre-Challenge Production setup

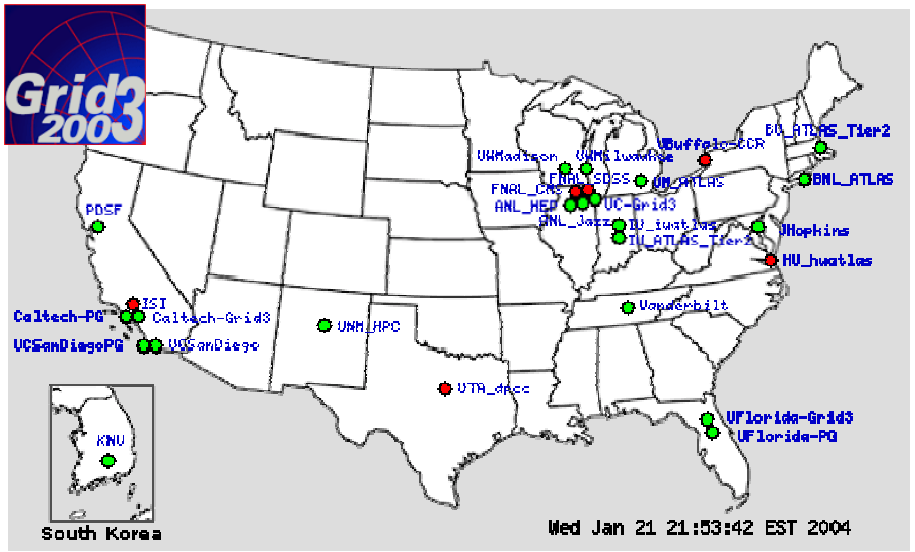
- CMS Production tools allow both local and Grid productions



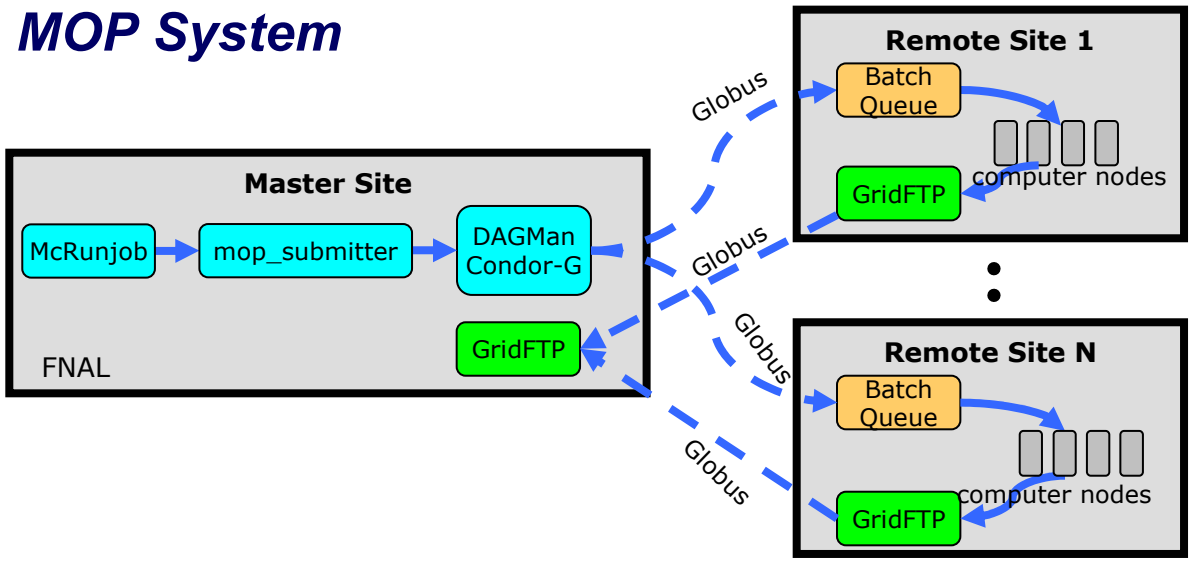
US MOP production system

Running on Grid2003

- Based on VDT1.1.11
- EDG VOMS for authentication
- GLUE Schema for MDS Information Providers
- MonaLisa for monitoring
- MOP for production control



MOP System

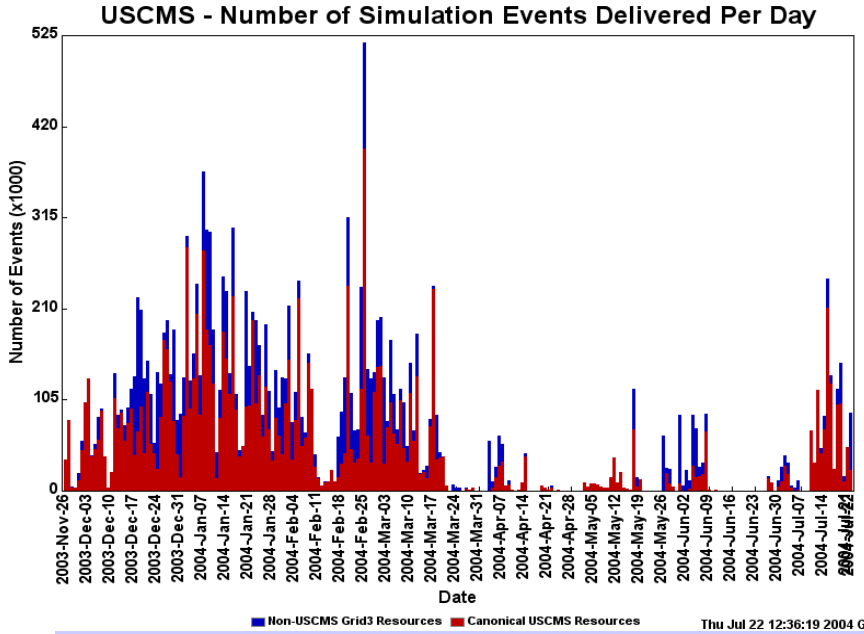


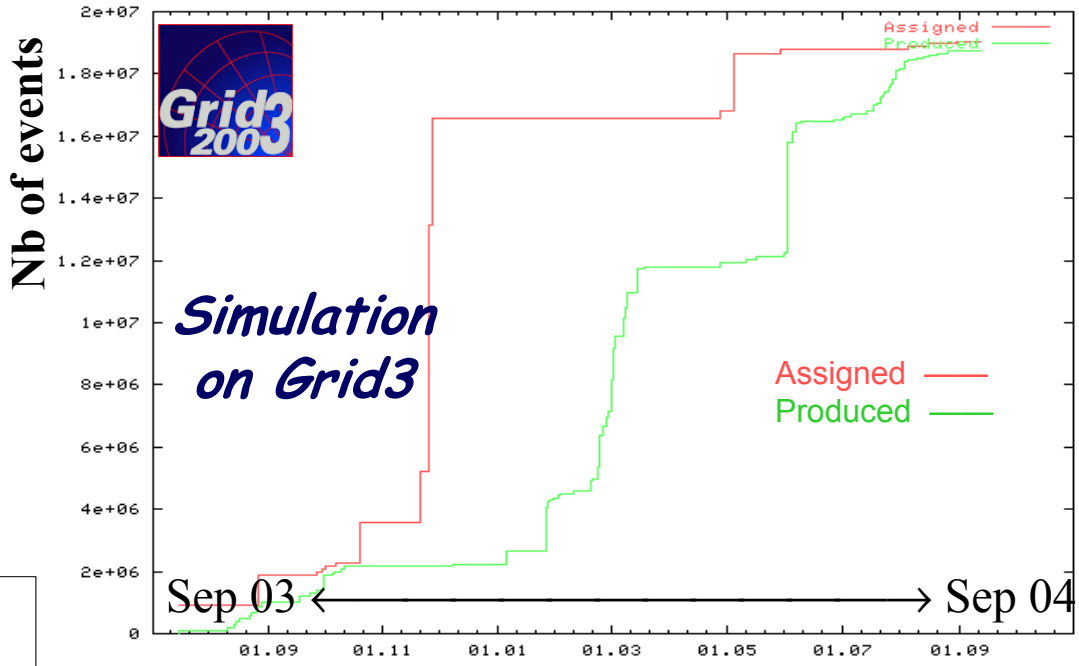
- Dagman and Condor-G for specification and submission
- Condor-based match-making process selects resources
- Results are returned using GridFTP to dCache at FNAL

Production on Grid: Grid3

Resources:

- US CMS Canonical resources (Caltech,UCSD,Florida,FNAL)
  - 500-600 CPUs
- Grid3 shared resources (~17 sites)
  - over 2000 CPUs (shared)
  - realistic usage (few hundred to 1000)





USMOP Regional Center Statistics:

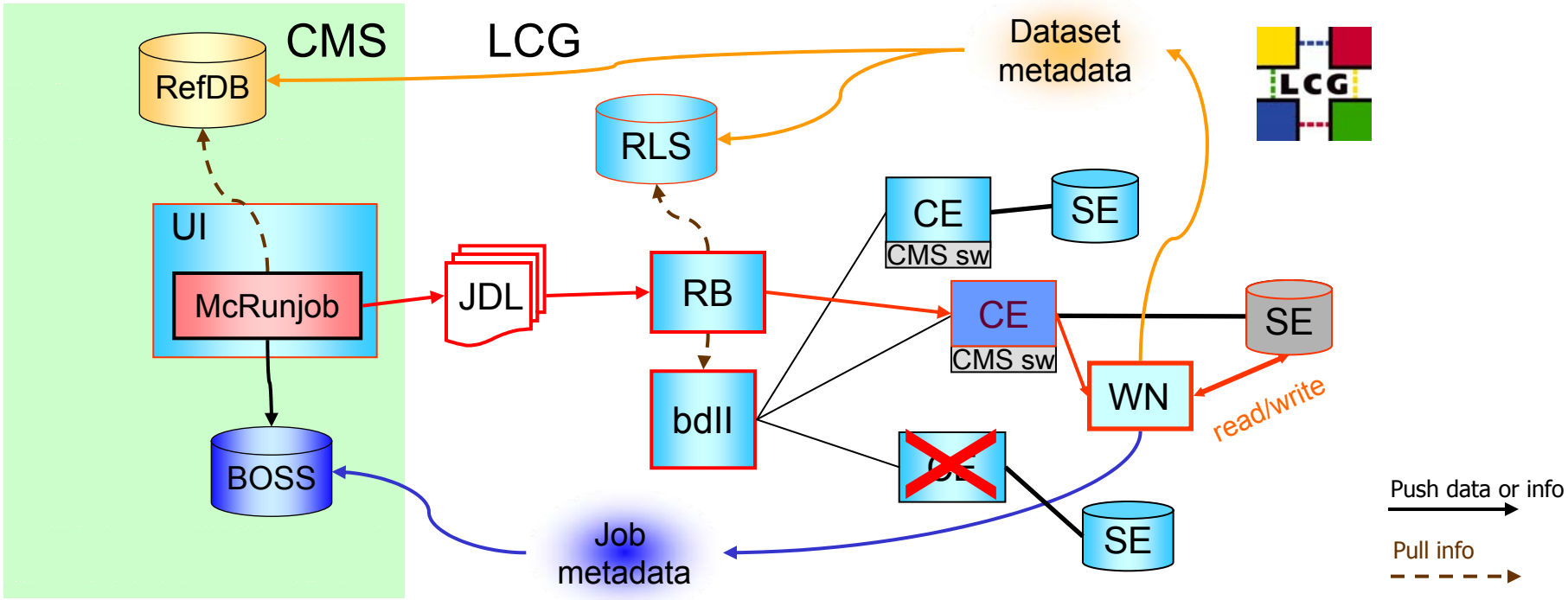
- 7.7 Mevts CMKIN
  - 30000 jobs ~ 0.7 KSI2000 months
- 19 Mevts CMSIM+OSCAR
  - 19000 jobs ~ 1000 KSI2K months
- 13 TB data

Grid3: results and observations

- Massive CMS Official Production on Grid3
  - ~ 19 Millions of events (19K very long jobs), 13TB data
  - Simultaneous usage of CPU resources peaked at 1200 CPUs, controlled by a single FTE
- Overall Job Efficiency ~ 70%
- Reasons of job failures
  - CMS application bugs ~ few %
  - No significant failure rate from Grid middleware per se
    - can generate high loads
    - infrastructure relies on shared filesystem
  - Most failures due to "normal" system issues
    - hardware failure
    - NIS, NFS problems
    - disks fill up
    - Reboots
  - Service level monitoring need to be improved
    - a service failure may cause all the jobs submitted to a site to fail
- The use of Grid-based jobs resulted in reducing the overall support effort required to submit and monitor jobs by a factor of two

CMS production interfaced to LCG

- Production is managed from User Interface
- CMS software installed on CE as RPM's



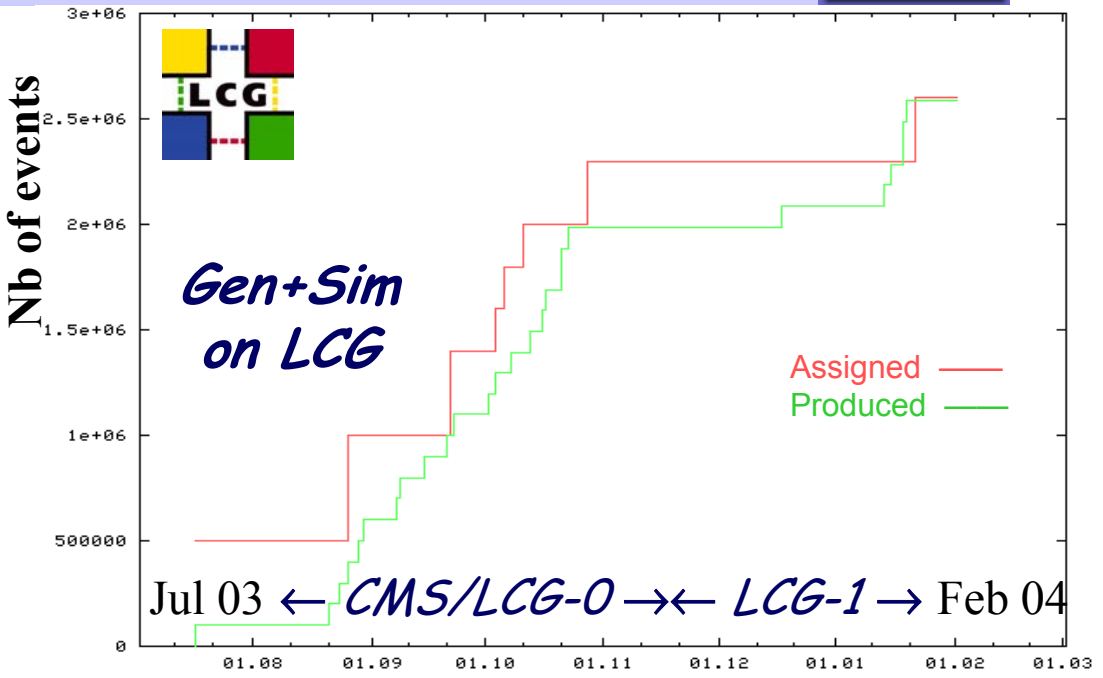
- Computing resources are matched by the Resource Broker to the job requirements (installed CMS software, MaxCPUTime, etc)
- Output data stored into close SE and registered in RLS

Production on grid: CMS-LCG

Resources:

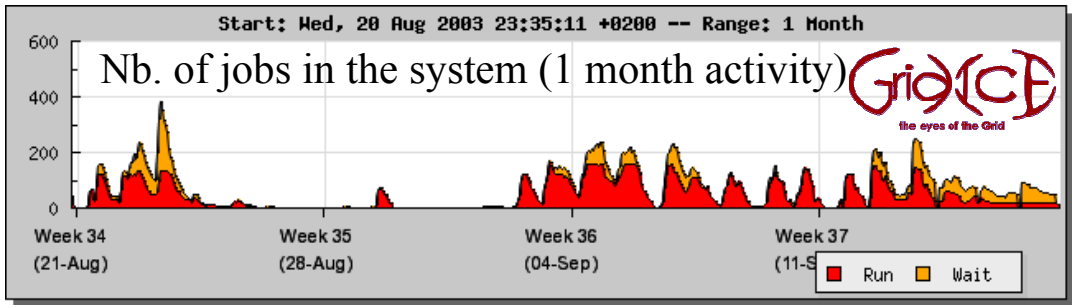
About 170 CPU's and 4TB

- CMS/LCG-0
  - CMS-wide testbed (~10 sites) based on the LCG pilot distribution (LCG-0) including RLS, VOMS, GLUE schema,GridICE…
- LCG-1
  - sites of "south testbed" (Italy-Spain)/Gridit



CMS-LCG Regional Center Statistics:

- 0.5 Mevts "heavy" CMKIN
  - 2000 jobs ~ 10 KSI2K months
- 2.1 Mevts CMSIM+OSCAR
  - 8500 jobs ~ 90 KSI2K months
- ~ 2 TB data



LCG: results and observations

- CMS Official Production on early deployed LCG implementations
  - ~ 2.6 Millions of events (~ 10K long jobs), 2TB data
- Overall Job Efficiency ranging from 70% to 90%
- The failure rate varied depending on the incidence of some problems:
  - RLS unavailability few times, in those periods the job failure rates could increase up to 25-30% → single point of failure
  - Instability due to site mis-configuration, network problems, local scheduler problem, hardware failure with overall inefficiency about 5-10%
  - Few % due to service failures
- Success Rate on LCG-1 was lower wrt CMS/LCG-0 (efficiency ~ 60%)
  - less control on sites, less support for services and sites (also due to Christmas)
  - Major difficulties identified in the distributed sites consistent configuration
- Good efficiencies and stable conditions of the system in comparison with what obtained in previous challenges
  - showing the maturity of the middleware and of the services, provided that a continuous and rapid maintenance is guaranteed by the middleware providers and by the involved site administrators

LCG-2 in CMS Data Challenge 04

Aspects of DC04 involving LCG-2 components

- register all data and metadata to a world-readable catalogue
  - RLS
- transfer the reconstructed data from Tier-0 to Tier-1 centers
  - Data transfer between LCG-2 Storage Elements
- analyze the reconstructed data at the Tier-1's as data arrive
  - Real-Time Analysis with Resource Broker on LCG-2 sites
- publicize to the community the data produced at Tier-1's
  - straightforward using the usual Replica Manager tools
- end-user analysis at the Tier-2's (not really a DC04 milestone)
  - first attempts
- monitor and archive resource and process information
  - GridICE

- Not a CPU challenge, but a full chain demonstration!
- Full chain (but the Tier-0 reconstruction) done in LCG-2

Description of CMS/LCG-2 system

- RLS at CERN with Oracle backend
- Dedicated information index (bdII) at CERN (by LCG)
  - CMS adds its own resources and removes problematic sites
- Dedicated Resource Broker at CERN (by LCG)
  - Other RB's available at CNAF and PIC, in future use them in cascade
- Official LCG-2 Virtual Organization tools and services
- Dedicated GridICE monitoring server at CNAF
- Storage Elements
  - Classic disk SE at CERN $\rightarrow$ Export Buffer
  - Castor SE at CNAF and PIC $\rightarrow$ import buffer from CERN and interface to MSS
  - Classic disk SE at CNAF, PIC, Legnaro, Ciemat $\rightarrow$ serve data for analysis
- Computing Elements at CNAF, PIC, Legnaro, Ciemat
  - CMS Software installed on CE by the CMS Software Manager via a grid job
    - RPM distribution based on CMSI
- User Interfaces at CNAF, PIC, LNL

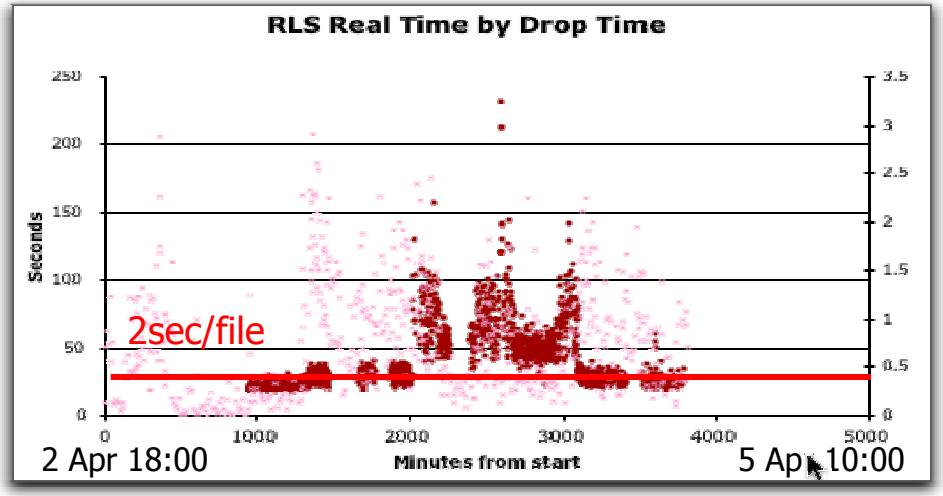
RLS usage

- CMS framework uses POOL
- RLS used as a global POOL catalogue, with full file meta data
  - Global file catalogue (LRC component of RLS: GUID ↔ PFNs)
    - Registration of files location by reconstruction jobs and by all transfer tools
    - Query by the Resource Broker to submit analysis jobs close to the data
  - Global metadata catalogue (RMC component of RLS: GUID ↔ metadata)
    - Meta data schema handled and pushed into RLS catalogue by POOL
    - Query (by users or agents) to find logical collection of files
    - CMS does not use a separate file catalogue for meta data
- Total Number of files registered in the RLS during DC04:
  - ~ 570K LFNs each with ~ 5-10 PFN's
  - 9 metadata attributes per file (up to ~1 KB metadata per file)

RLS issues

- Inserting information into RLS:
  - insert PFN (file catalogue) was fast enough if using the appropriate tools, produced in-course
    - LRC C++ API programs (~0.1-0.2sec/file), POOL CLI with GUID (secs/file)
  - insert files with their attributes (file and metadata catalogue) was slow
    - We more or less survived, higher data rates would be troublesome
- Querying information from RLS
  - Looking up file information by GUID seems sufficiently fast
  - Bulk queries by GUID take a long time (seconds per file)
  - Queries on metadata are too slow (hours for a dataset collection)

Time to register the output of a Tier-0 job (16 files)

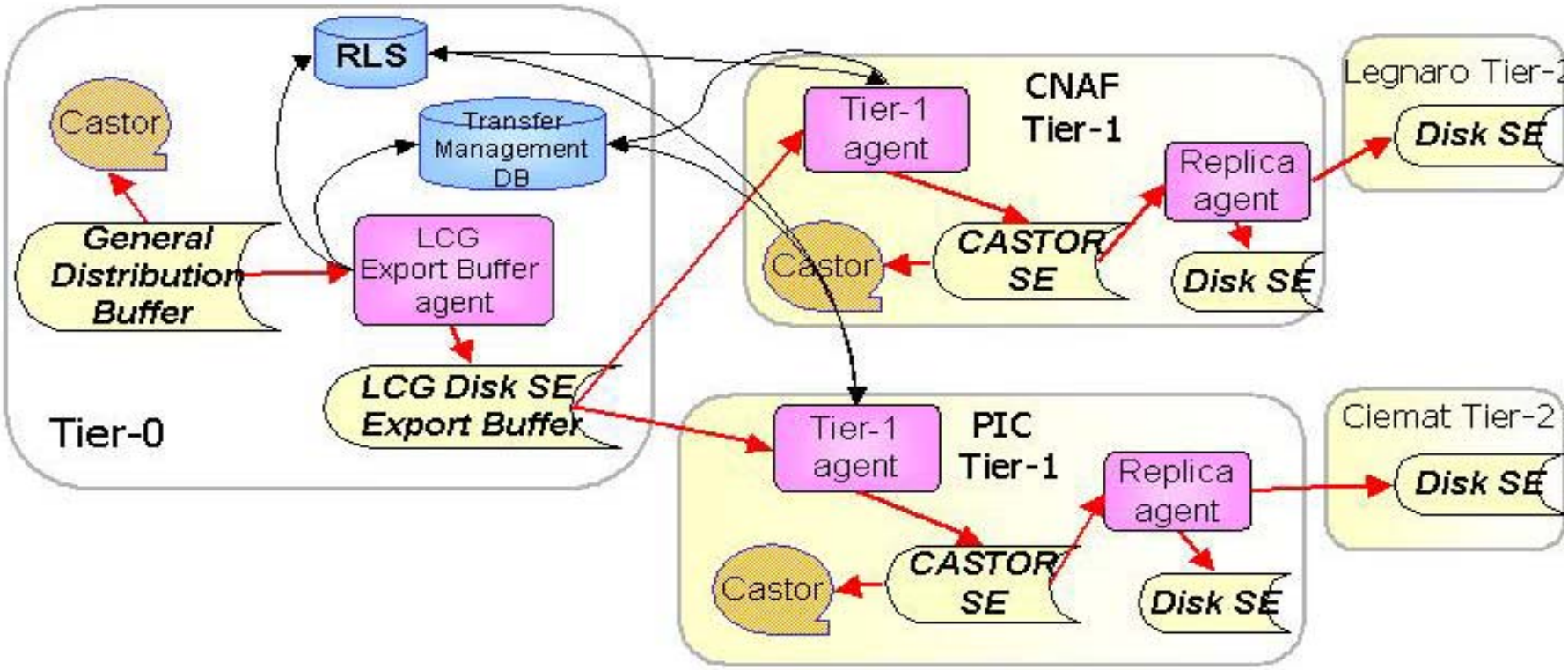


Sometimes the load on RLS increases and requires intervention on the server (i.e. log partition full, switch of server node, un-optimized queries)

⇒ able to keep up in optimal condition, so and so otherwise

Data Transfer (I)

- Set of agents communicating through the Transfer Management DB (TMDB)



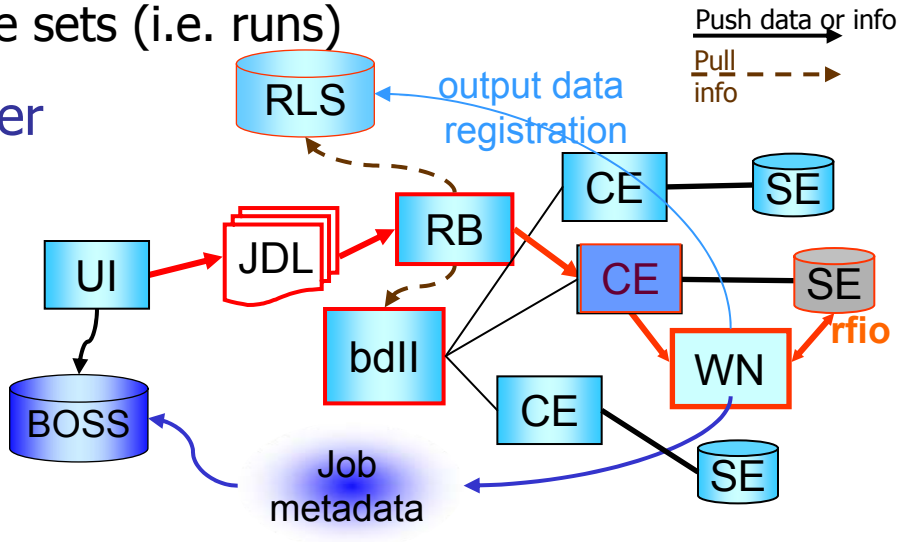
- Data upload at Tier-0 in a disk SE Export Buffer and register in RLS
- Data transfer from Tier-0 to CASTOR SEs at Tier-1
- Data replication from Tier-1 to Tier-2 disk SEs

Data Transfer (II)

- Transfer tools:
  - Replica Manager CLI used for EB ➔ CNAF and CNAF ➔ Legnaro
    - Java-based CLI introduces non negligible overhead at start-up
  - globus-url-copy + LRC C++ API used for EB ➔PIC and PIC ➔ Ciemat
    - Faster
- Performance has been good with both tools
  - able to keep up with the rate of data coming from the reconstruction at Tier-0
  - Total network throughput limited by small file size
  - Some transfer problem caused by performance of underlying MSS (Castor)

Real-time Data Analysis

- Automatic procedures to submit analysis jobs as new data were made available on disk SE at Tier-1 and Tier-2
  - Main difficulty is to identify complete file sets (i.e. runs)
- Job submission to LCG-2 Resource Broker
  - running on LCG-2 sites (Spain and Italy Tier-1/2)
  - Job sent close to the data
    - File access via rfio
  - Output data registered in RLS
  - Job monitoring with BOSS



- The LCG submission system could cope with the rate of data coming from CERN
  - More than 17000 analysis jobs were submitted in about 2 weeks, with a grid efficiency of 90-95%
  - During the last days of running an average delay of 20 minutes from data at Tier-0 to their analysis at Tier-1 was measured
- Real-time analysis sustained running was done only in LCG environment

Conclusions

- CMS distributed production based on grid middleware used within the official CMS production system
  - Grid3: reliable and scalable system for massive production
  - LCG: large scale productions proved
    - distributed sites consistent configuration and control is very important
- CMS Data Challenge
  - LCG environment provides the functionalities for distributed computing
    - The catalogues are an issue!
    - Grid point-to-point file transfer tools
    - Infrastructure for data analysis
  - LCG data distribution and data analysis chain successfully met the data challenge goals of large scale scheduled distribution to a set of Tier-1/2 and subsequent analysis